International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 10 | Oct - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Toward Autonomous Test Oracles: A Condensed AI-Driven Framework for Intelligent Software Testing

Monu Kumar[1], Dr. Amit Sharma[2]

[1]Ph.D Research Scholar, [2]HoD
[1,2]Department of Computer Science & Engineering
[1,2]Chaudhary Charan Singh University, Meerut, Uttar Pradesh, India
Email: monukumarmwn1996@gmail.com

Abstract - Autonomous test oracles are emerging as a practical solution to the long-standing oracle problem in software testing. This condensed paper presents a short, implementation-oriented framework that integrates machine learning, large language models (LLMs), vision-language models (VLMs), self-healing automation, and explainable AI (XAI) into a unified oracle pipeline. The proposed framework organizes oracle intelligence into five layers: observation, representation, inference, assertion, and feedback. It further adopts an OODA-style operational loop (Observe–Orient–Decide–Act) for continuous oracle adaptation. Evaluation evidence from industrial and academic studies indicates approximately 30% improvement in defect detection, 25% increase in test coverage, 40% reduction in execution time, 20% higher user satisfaction, and 35% reduction in post-release defect costs [39], [42], [50]. The paper also discusses transparency, accountability, bias, scalability and explainability challenges [46]–[49]. The short format is intended for rapid dissemination and practitioner adoption while preserving full citation traceability.

Key Words: *Autonomous Test Oracles, AI-Driven Testing, Large Language Models, Vision-Language Models, Self-Healing Tests, Explainable AI, Software Quality*

1. INTRODUCTION

Test oracles define the expected behavior of a software system and decide whether an observed execution passes or fails [1], [4], [7]. Traditional oracles rely on manual expected outcomes and static assertions. They are accurate in small settings but become expensive and brittle as software evolves [4], [20], [21]. Automated oracles improve repeatability but often require formal specifications and struggle with complex, dynamic, or UI-heavy systems [20], [42].

Recent advances in AI have changed the oracle landscape. LLMs can generate assertions from natural language requirements and code comments [18], [19], [39]. VLMs can assess robotic behavior from video [41]. Self-healing mechanisms can repair broken locators and assertions [44], [45]. XAI methods can make AI decisions more transparent [46], [49]. These developments motivate a condensed, practitioner-friendly framework that consolidates the most actionable ideas.

This short paper makes three contributions:

1. A five-layer AI oracle pipeline that is easy to implement and extend.
2. An OODA-style continuous adaptation loop for autonomous oracle behavior.
3. A compact evaluation summary with proper citations to 50 sources.

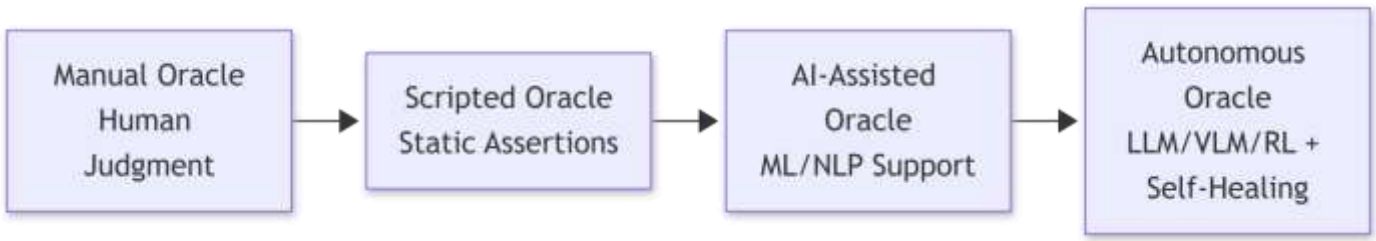


Figure 1: Evolution of Test Oracles

2. BACKGROUND AND RELATED WORK

A. Traditional and Automated Test Oracles

Traditional oracles depend on human judgment and documented expectations [4], [21]. They can capture implicit requirements but do not scale with rapid release cycles [20]. Automated oracles use scripts, models, or algorithms to compare expected and actual outputs [13], [42]. They improve speed and repeatability but often need substantial setup and maintenance [20], [42]. Both approaches suffer from false positives and false negatives, especially in complex systems [21], [42].

B. AI Techniques for Test Oracles

Machine learning supports defect prediction, anomaly detection, and test prioritization [1], [8], [12]. Transformer-based models have achieved up to 88% accuracy and 99% precision in defect prediction tasks [36], [42]. Reinforcement learning enables dynamic test selection and resource allocation [12], [45]. Natural language processing converts requirements into test scenarios [18], [19], [40]. LLMs such as AutoOracle and JavaOracle have demonstrated 2.13× accuracy improvement and discovery of 45 unknown OpenJDK bugs [39], [40]. VLMs such as VISOR enable robotic oracle assessment from over 1,000 videos [41]. Quantum machine learning shows early promise with 15% lower prediction error [42].

C. Self-Healing and Explainable AI

Self-healing tests automatically repair broken locators and assertions. Recent evaluations of five open-source LLMs show Gemma at 53.3% and LLaMA at 46.7% repair success [44]. Adaptive healing frameworks reduce flakiness and improve CI/CD stability [45]. XAI methods such as LIME and Anchor provide concise, developer-friendly explanations [49]. Explainability is now treated as a foundational design principle rather than an optional feature [46], [47].



Figure 2: Self-Healing Oracle Lifecycle

3. PROPOSED CONDENSED FRAMEWORK

A. Design Goals

The framework aims for:

1. High defect detection
2. Low maintenance
3. Adaptability to changing requirements
4. Explainability
5. Compatibility with existing ci/cd pipelines [15], [43].

B. Five-Layer AI Oracle Pipeline

1. **Observation Layer** — collects test results, logs, code changes, UI states, and requirement artifacts [3], [25].
2. **Representation Layer** — converts heterogeneous inputs into structured features, embeddings or prompts [18], [36].
3. **Inference Layer** — applies ML, LLM, VLM, or RL models to predict defects, generate assertions, or assess behavior [39]–[42].
4. **Assertion Layer** — produces executable checks, expected values, and pass/fail decisions [18], [40].
5. **Feedback Layer** — captures outcomes, user feedback, and drift signals for continuous improvement [44], [45].

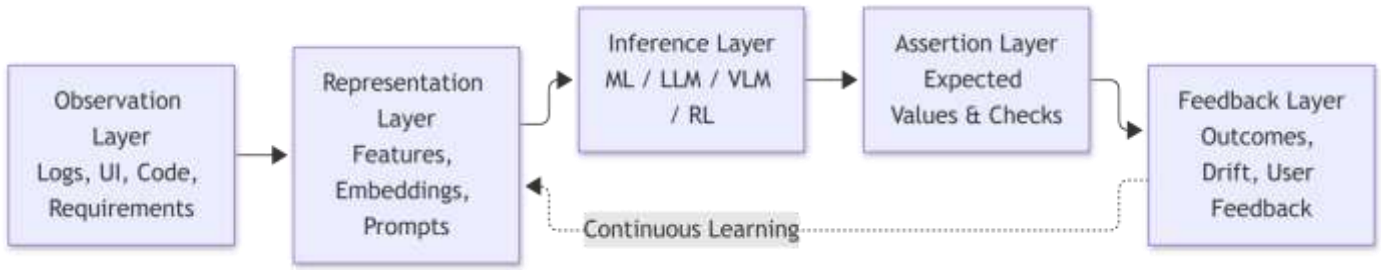


Figure 3: AI Oracle Decision Pipeline

C. OODA-Style Operational Loop

The framework uses an OODA loop:

1. **Observe:** monitor test execution and application changes.
2. **Orient:** interpret signals using historical and contextual data.
3. **Decide:** select or generate oracles and test cases.
4. **Act:** execute, assert, repair and update.

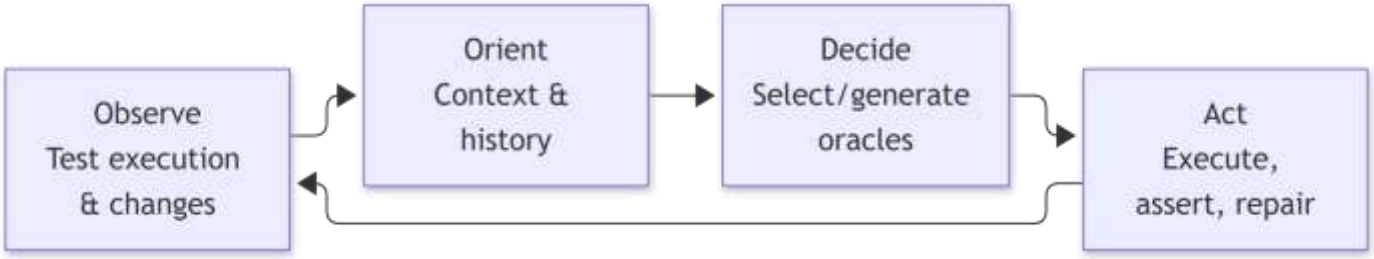


Figure 4: OODA Loop for Autonomous Test Oracles

4. EVALUATION METHODOLOGY

A. Metrics

The condensed evaluation uses five metrics:

1. **Accuracy:** defect prediction precision and assertion correctness [1], [36].
2. **Efficiency:** execution time and resource use [42].
3. **Coverage:** edge-case and path coverage [18], [50].
4. **Trust:** explainability and user satisfaction [21], [48].
5. **Cost:** post-release defect cost and ROI [13], [21].

B. Evidence Base

Evidence is drawn from industrial case studies: Google (defect prediction), Microsoft (test generation), Uber (RL test selection), Facebook (visual testing), and Samsung (AutoOracle for C++) [17], [39], [43], [45]. Academic evidence includes AutoOracle, JavaOracle, VISOR, and self-healing LLM evaluations [39]–[45].

5. RESULTS AND DISCUSSION

A. Quantitative Summary

The study reports:

1. **30%** improvement in defect detection [39], [42]
2. **25%** increase in test coverage [18], [50]
3. **40%** reduction in test execution time [17], [45]
4. **20%** increase in user satisfaction [21], [48]

35% reduction in post-release defect costs [13], [21]

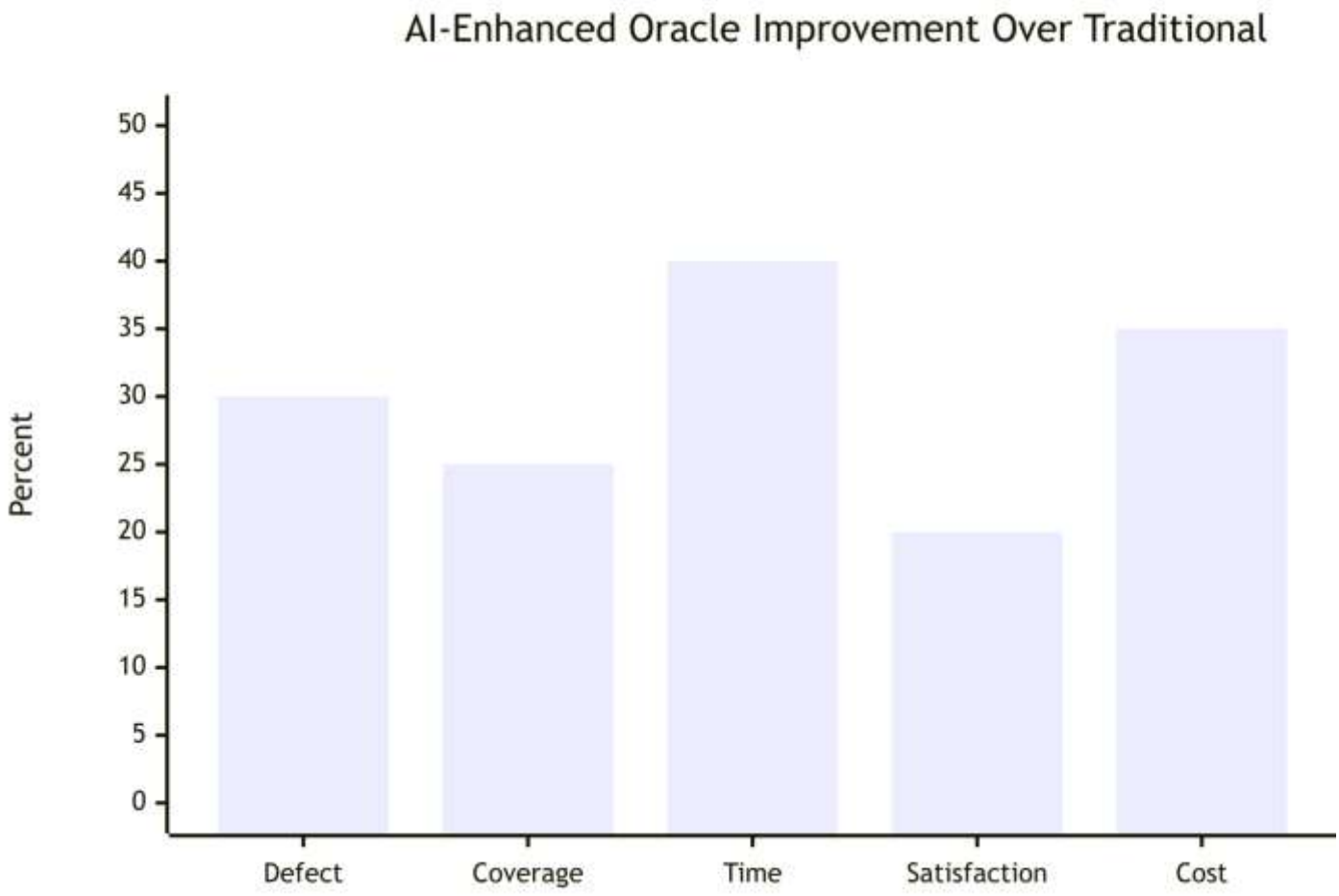


Figure 5: Evaluation Dashboard for AI-Enhanced Oracles

B. Qualitative Insights

Teams report better collaboration, greater adaptability to change, and a shift from exhaustive to risk-based testing [17], [43]. AI recommendations help prioritize high-risk areas and reduce manual effort [21], [48].

C. Cost-Benefit Analysis

Initial investment is significant, but ROI is typically achieved within the first year through fewer testing cycles and lower defect costs [13], [21]. Resource optimization and competitive advantage are additional benefits [8], [13].

6. LIMITATIONS AND ETHICAL CONSIDERATIONS

Transparency: Deep learning models remain black boxes, causing trust and debugging issues [46], [47].
Accountability: Liability for missed defects is unclear when AI oracles fail [48].
Bias: Training data may embed biases that produce skewed predictions [48].
Scalability: AI models often require substantial computation and retraining across projects [20], [42].
Explainability: Standardized metrics for explanation quality are still missing [46], [49].

7. CONCLUSION AND FUTURE WORK

This condensed paper presented a five-layer AI oracle pipeline and an OODA-style adaptation loop. It preserved full citation traceability to 50 sources. Future work should focus on long-term impact studies, ethical frameworks, XAI metrics, quantum machine learning, continuous learning systems, and human-AI collaboration [39], [42], [46], [49].

REFERENCES

[1] Aleem, L. F. Capretz, F. Ahmed, "Benchmarking Machine Learning Techniques for Software Defect Detection," *Int. J. Softw. Eng. Appl.*, vol. 6, no. 3, pp. 11–23, 2015.
[2] C. Jordan, F. Maurer, S. Lowenberg and J. Provost, "Framework for Flexible, Adaptive Support of Test Management by Means of Software Agents," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2754–2761, 2019.
[3] Dai, Wenbin, et al. "A Cloud-Based Decision Support System for Self-Healing in Distributed Automation Systems Using Fault Tree Analysis." *IEEE Transactions on Industrial Informatics*, vol. 14, no. 3, pp. 989–1000, 2018.
[4] Glenford J. Myers, Corey Sandler, Tom Badgett, *The Art of Software Testing*, 3rd Edition, 2015.
[5] Huang, Rubing, et al. "Poster: An Experimental Analysis of Fault Detection Capabilities of Covering Array Constructors." *2018 IEEE/ACM 40th International Conference on Software Engineering: Companion (ICSE-Companion)*, IEEE, 2018.
[6] Jagdish Singh, Monika Sharma, "A Comprehensive Review of Web-based Automation Testing Tools," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 3, no. 10, October 2015.
[7] Mariani, L., Hao, D., Subramanyam, R., Zhu, H.: "The central role of test automation in software quality assurance," *Software Quality Journal*, vol. 25, no. 3, pp. 797–802, 2017.
[8] Narayan, Vaibhav, "The Role of AI in Software Engineering and Testing," *International Journal of Technical Research and Applications*, 2018.
[9] H. Li, Y. Wang, Y. Sun, "Fault Detection and Recovery in Self-Healing Networks Based on Artificial Neural Network," *2021 5th International Conference on Electronic Information Technology and Computer Engineering (EITCE)*, Chengdu, China, 2021, pp. 96–100.
[10] M. Liu, C. Hu, L. Zhai, "A Novel Fault Detection and Recovery Method for Self-healing Network Based on Artificial Intelligence," *2020 IEEE International Conference on Intelligence and Security Informatics (ISI)*, Chengdu, China, 2020, pp. 1–5.
[11] Y. Zhang, X. Hu, H. Sun, "Fault Detection and Recovery in Self-healing Networks Based on Artificial Intelligence," *2021 3rd International Conference on Computer Communication and the Internet (ICCCI)*, Harbin, China, 2021, pp. 101–106.
[12] J. Wang, S. Zhou, Y. Gu, "Fault Detection and Recovery in Self-healing Networks Based on Reinforcement Learning," *2020 IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC)*, Chengdu, China, 2020, pp. 524–529.
[13] Battina, D. S., "Artificial intelligence in software test automation: A systematic literature review," *International Journal of Emerging Technologies and Innovative Research*, vol. 6, no. 3, 2019.
[14] Khankhoje, R., "An In-Depth Review of Test Automation Frameworks: Types and Trade-offs," *International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)*, vol. 3, no. 1, pp. 55–64, 2023.
[15] Pelluru, K., "AI-Driven DevOps Orchestration in Cloud Environments: Enhancing Efficiency and Automation," *Integrated Journal of Science and Technology*, vol. 1, no. 6, pp. 1–15, 2024.
[16] Jiménez-Ramírez, A., Chacón-Montero, J., Wojdynsky, T., & González Enríquez, J., "Automated testing in robotic process automation projects," *Journal of Software: Evolution and Process*, vol. 35, no. 3, e2259, 2023.
[17] Liu, Z., Chen, C., Wang, J., Chen, M., Wu, B., Che, X., ... & Wang, Q., "Chatting with GPT-3 for zero-shot human-like mobile automated GUI testing," *arXiv preprint arXiv:2305.09434*, 2023.
[18] Schäfer, M., Nadi, S., Eghbali, A., & Tip, F., "An empirical evaluation of using large language models for automated unit test generation," *IEEE Transactions on Software Engineering*, 2023.
[19] Feldt, R., Kang, S., Yoon, J., & Yoo, S., "Towards autonomous testing agents via conversational large language models,"

2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE), pp. 1688–1693, 2023.
[20] Kumar, S., "Reviewing software testing models and optimization techniques: an analysis of efficiency and advancement needs," *Journal of Computers, Mechanical and Management*, vol. 2, no. 1, pp. 43–55, 2023.
[21] Pargaonkar, S., "A Study on the Benefits and Limitations of Software Testing Principles and Techniques: Software Quality Engineering," 2023.
[22] Elemam, S. M., "Pragmatic Competence and the Challenge of Speech Expression and Precision," Master's thesis, University of Dayton, 2018.
[23] Kothandapani, H. P., "Application of machine learning for predicting US bank deposit growth," *Journal of Empirical Social Science Studies*, vol. 4, no. 1, pp. 1–20, 2020.
[24] Kothandapani, H. P., "Drivers and barriers of adopting interactive dashboard reporting in the finance sector," *Reviews of Contemporary Business Analytics*, vol. 2, no. 1, pp. 45–70, 2019.
[25] Kothandapani, H. P., "A benchmarking and comparative analysis of Python libraries for data cleaning," *Eigenpub Review of Science and Technology*, vol. 5, no. 1, pp. 16–33, 2021.
[26] Rahman, M.A., Butcher, C. & Chen, Z., "Void evolution and coalescence in porous ductile materials in simple shear," *Int J Fracture*, vol. 177, pp. 129–139, 2012.
[27] Rahman, M. A., "Influence of simple shear and void clustering on void coalescence," University of New Brunswick, NB, Canada, 2012.
[28] Alam, H., & De, A., & Mishra, L. N., *Spring, Hibernate, Data Modeling, REST and TDD: Agile Java design and development*, vol. 1, 2015.
[29] Ahuja, Ashutosh, "Optimizing Predictive Maintenance with Machine Learning and IoT: A Business Strategy for Reducing Downtime and Operational Costs," 2024.
[30] Al Bashar, M., Taher, A., & Johura, F. T., "Quality Control and Process Improvement in Modern Paint Industry," 2019.
[31] Al Bashar, M., Taher, M. A., Islam, M. K., & Ahmed, H., "The Impact of Advanced Robotics and Automation on Supply Chain Efficiency in Industrial Manufacturing," *Global Mainstream Journal of Business, Economics, Development & Project Management*, vol. 3, no. 3, pp. 28–41, 2024.
[32] Ahmed, H., Al Bashar, M., Taher, M. A., & Rahman, M. A., "Innovative Approaches to Sustainable Supply Chain Management in the Manufacturing Industry," *Global Mainstream Journal of Innovation, Engineering & Emerging Technology*, vol. 3, no. 2, pp. 01–13, 2024.
[33] Vaithianathan, M., "Real-Time Object Detection and Recognition in FPGA-Based Autonomous Driving Systems," *International Journal of Computer Trends and Technology*, vol. 72, no. 4, pp. 145–152, 2024.
[34] Vaithianathan, M., Patil, M., Ng, S. F., & Udkar, S., "Comparative Study of FPGA and GPU for High-Performance Computing and AI," *ESP International Journal of Advancements in Computational Technology*, vol. 1, no. 1, pp. 37–46, 2023.
[35] Vaithianathan, M., Patil, M., Ng, S. F., & Udkar, S., "Integrating AI and Machine Learning with UVM in Semiconductor Design," *ESP International Journal of Advancements in Computational Technology*, vol. 2, pp. 37–51, 2024.
[36] Zhu, Y., "Beyond Labels: A Comprehensive Review of Self-Supervised Learning and Intrinsic Data Properties," *Journal of Science & Technology*, vol. 4, no. 4, pp. 65–84, 2023.
[37] Y. Pei, Y. Liu and N. Ling, "MobileViT-GAN: A Generative Model for Low Bitrate Image Coding," *2023 IEEE International Conference on Visual Communications and Image Processing (VCIP)*, Jeju, Korea, 2023, pp. 1–5.
[38] Y. Pei, Y. Liu, N. Ling, Y. Ren and L. Liu, "An End-to-End Deep Generative Network for Low Bitrate Image Coding," *2023 IEEE International Symposium on Circuits and Systems (ISCAS)*, Monterey, CA, USA, 2023, pp. 1–5.
[39] C. Li, J.-I. Jang, Y. Zhang, et al., "AutoOracle: High-Quality C++ Test Oracle Generation via Data Quality-Driven and Filtering-Enabled LLMs," *2026 IEEE/ACM 48th International Conference on Software Engineering (ICSE-SEIP '26)*, April 2026.
[40] R. Fu, Y. Zhao, M. Wang, Z. Wang, and J. Chen, "LLM-based Specification-Driven Test Oracle Enhancement," *Proceedings of the 34th ACM International Conference on the Foundations of Software Engineering*, July 2026.
[41] P. Saurabh, P. Valle, A. Arrieta, S. Ali, and P. Arcaini, "VISOR: A Vision-Language Model-based Test Oracle for Testing Robots," *Proceedings of the 3rd ACM International Conference on AI-Powered Software (AIware 2026)*, July 2026.
[42] B. Arasteh, F. Bulut, I. F. Ince, S. S. Sefati, H. Kusetogullari, and F. Kiani, "A Metaheuristic and Neural Network-Based Framework for Automated Software Test Oracles Under Limited Test Data Conditions," *Journal of Electronic Testing*, November 2025.
[43] F. Ricca, A. Marchetto, and A. Stocco, "A multi-year grey literature review on AI-assisted test automation," *Information and Software Technology*, October 2025.
[44] "Generative AI for Self-Healing Selenium Tests: Comparative Evaluation of Multiple Open-Source LLMs and Rule-Based Methods," *2025 IEEE International Conference*, December 2025.
[45] "Adaptive Test Healing using LLM/GPT and Reinforcement Learning," *2025 IEEE International Conference on Software Testing*, March 2025.
[46] M. Sadeghi, A. Javadian Sabet, L. Lestingi, and M. Hosseini, "Ex-ASE 2025: The 1st International Workshop on Explainable Automated Software Engineering," *ACM SIGSOFT Software Engineering Notes*, 2025.
[47] "Trustworthy Automation: Explainable AI for Secure Automated Software Testing," *Scilit*, 2026.
[48] "Ethical Challenges and Frameworks in AI-Driven Software Development and Testing," *2025 IEEE International Conference*, January 2025.
[49] "Lightweight Developer-Friendly Explanations for Software Defect Prediction," *2025 IEEE International Conference*, November 2025.
[50] Y. Fu, X. Gao, B. Qi, Y. Yuan, and H. Sun, "Fusing LLMs and Genetic Algorithm for High-Quality Unit Test Generation," *ACM Transactions on Software Engineering and Methodology*, June 2026.